ON THE VARIANCE OF THE ADAPTIVE LEARNING RATE AND BEYOND

Liyuan Liu [*]
University of Illinois, Urbana-Champaign
`ll2@illinois`

Haoming Jiang [†]
Georgia Tech
`jianghm@gatech.edu`

Pengcheng He, Weizhu Chen
Microsoft Dynamics 365 AI
`{penhe,wzchen}@microsoft.com`

Xiaodong Liu, Jianfeng Gao
Microsoft Research
`{xiaodl,jfgao}@microsoft.com`

Jiawei Han
University of Illinois, Urbana-Champaign
`hanj@illinois`

ABSTRACT

The learning rate warmup heuristic achieves remarkable success in stabilizing training, accelerating convergence and improving generalization for adaptive stochastic optimization algorithms like RMSprop and Adam. Here, we study its mechanism in details. Pursuing the theory behind warmup, we identify a problem of the adaptive learning rate (*i.e.*, it has problematically large variance in the early stage), suggest warmup works as a variance reduction technique, and provide both empirical and theoretical evidence to verify our hypothesis. We further propose RAdam, a new variant of Adam, by introducing a term to rectify the variance of the adaptive learning rate. Extensive experimental results on image classification, language modeling, and neural machine translation verify our intuition and demonstrate the effectiveness and robustness of our proposed method. [1]

1 INTRODUCTION

Fast and stable optimization algorithms are what generations of researchers have been pursuing (Cauchy, 1847). Remarkably, stochastic gradient-based optimization, such as stochastic gradient descent (SGD), has witnessed tremendous success in many fields of science and engineering despite its simplicity. Recently, many new methods have been proposed to accelerate optimization by applying *adaptive learning rate*. In particular, Adagrad (Duchi et al., 2011) and its variants, *e.g.*, RMSprop (Tieleman & Hinton, 2012), Adam (Kingma & Ba, 2014), Adadelta (Zeiler, 2012) and Nadam (Dozat, 2016), have been widely used due to their fast convergence.

However, it has been observed that in many cases, these optimization methods converge to bad/suspicious local optima, and have to resort to a warmup heuristic – using a small learning rate in the first few epochs of training – to mitigate the convergence problem (Vaswani et al., 2017; Popel & Bojar, 2018). For example, on the De-En IWSLT’14 dataset, removing warmup increases the training perplexity of a Transformer language model from 10 to over 500, as shown in Figure 1. Similar phenomena are observed in other scenarios like BERT pre-training (Devlin et al., 2018).

Since the theoretical underpinnings of the warmup heuristic are lacking, there is neither guarantee that it always works in various machine learning settings nor guidance on how we should conduct warmup. Thus, researchers typically use different settings in different applications and have to take a trial-and-error approach, which can be tedious and time-consuming.

In this paper, we conduct both theoretical and empirical analysis of the convergence issue to identify its origin. Specifically, we show that its root cause is that the adaptive learning rate has undesirably large variance in the early stage of model training due to the limited amount of training samples being used. Thus, to reduce such variance, it is better to use smaller learning rates in the first few epochs of training, which justifies the warmup heuristic.

[*] Work was done during an internship at Microsoft.

[†] Work was done during an internship at Microsoft.

[1] All implementations are available at: `https://github.com/LiyuanLucasLiu/RAdam`.

Moreover, we propose a new variant of Adam, called Rectified Adam (RAdam), which explicitly rectifies the variance of the adaptive learning rate based on derivations. We conduct extensive experiments on language modeling, image classification, and neural machine translation. RAdam brings consistent improvement over the vanilla Adam, which verifies the variance issue generally exists on various tasks across different network architectures.

Our main contributions are two-fold:

- We identify the variance issue of the adaptive learning rate and present a theoretical justification for the warmup heuristic. We show that the convergence issue is due to the undesirably big variance of the adaptive learning rate in the early stage of model training.

- We propose a new variant of Adam (*i.e.*, RAdam), which not only explicitly rectifies the variance and is theoretical sound, but also compares favorably with the heuristic warmup.

2 Preliminaries and Motivations

Generic adaptive methods. Algorithm 1 is a generic framework (all operations are element-wise) that describes many popular adaptive gradient descent methods algorithms (Reddi et al., 2019). Specifically, different optimization algorithms can be specified by different choices of $\phi(.)$ and $\psi(.)$. For example, in the Adam algorithm, these two functions are set to:

$$\phi(g_1, \cdots, g_t) = \frac{(1-\beta_1)\sum_{i=1}^{t}\beta_1^{t-i}g_t}{1-\beta_1^t} \quad \text{and} \quad \psi(g_1, \cdots, g_t) = \sqrt{\frac{1-\beta_2^t}{(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}g_i^2}}. \tag{1}$$

For numerical stability, $\psi(.)$ is usually calculated as $\widehat{\psi}(g_1, \cdots, g_t) = \frac{\sqrt{1-\beta_2^t}}{\epsilon+\sqrt{(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}g_i^2}}$, where ϵ is set to a relatively small value (*e.g.*, 1×10^{-8}).

Algorithm 1: Generic adaptive optimization method setup. All operations are element-wise.

```
Input: $\{\alpha_t\}_{t=1}^T$: step size, $\{\phi_t, \psi_t\}_{t=1}^T$: function to calculate momentum and adaptive rate,
       $\theta_0$: initial parameter, $f(\theta)$: stochastic objective function.
Output: $\theta_T$: resulting parameters
while $t = 1$ to $T$ do
    $g_t \leftarrow \Delta_\theta f_t(\theta_{t-1})$ (Calculate gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \phi_t(g_1, \cdots, g_t)$ (Calculate momentum)
    $v_t \leftarrow \psi_t(g_1, \cdots, g_t)$ (Calculate adaptive learning rate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha_t m_t v_t$ (Update parameters)
return $\theta_T$
```

Learning rate warmup. Instead of setting the learning rate α_t as a constant or in decreasing order, a learning rate warmup strategy sets α_t as some small values in the first few steps. For example, linear warmup sets $\alpha_t = t\,\alpha_0$ when $t < T_w$. Warmup has been demonstrated to be beneficial in many deep learning applications. For example, in the NMT experiments in Figure 1, the perplexity convergences around 500 when warmup is not applied (Adam-vanilla), and it surprisingly decreases to below 10 after applying warmup (Adam-warmup).

To further analyze this phenomenon, we visualize the histogram of the absolute value of gradients on a log scale in Figure 2. We observe that, without applying warmup, the gradient distribution is distorted to have a mass center in relatively small values within 10 updates. Such gradient distortion means that the vanilla Adam is trapped in bad/suspicious local optima after the first few updates. Warmup essentially reduces the impact of these problematic updates to avoid the convergence problem. In the following sections, we focus our analysis on learning rate warmup for the Adam algorithm, while it can be applied to other algorithms that use similar adaptive learning rate ($\psi(.)$) designs, *e.g.*, RMSprop (Tieleman & Hinton, 2012) and Nadam (Dozat, 2016).

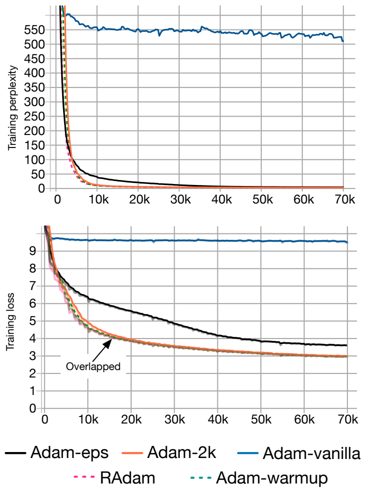


Figure 1: Training of Transformers on the De-En IWSLT'14 dataset. Up: Training perplexity w.r.t. gradient update iterations; Bottom: Training loss w.r.t. gradient update iterations.

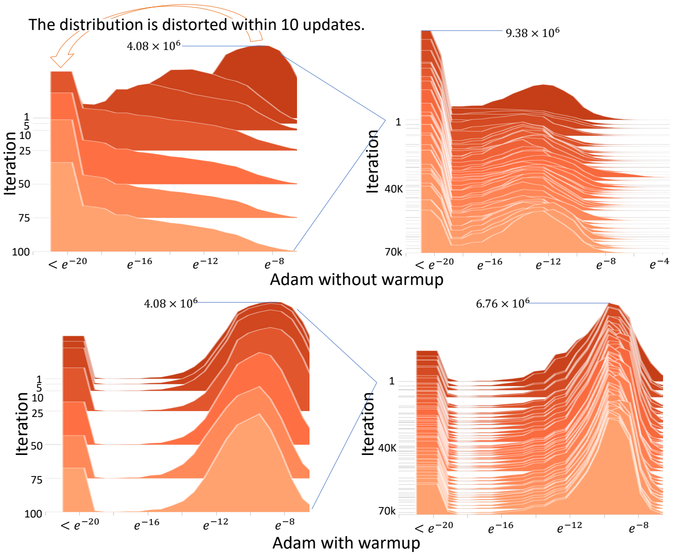


Figure 2: The absolute gradient histogram of the Transformers on the De-En IWSLT' 14 dataset. X-axis is absolute value in the log scale and the height is the frequency. Without warmup, the gradient distribution is distorted in the first 10 steps.

3 Variance of Adaptive Rate

In this section, we first introduce empirical evidence, then analyze the variance of the adaptive learning rate to support our hypothesis – *Due to the lack of samples in the early stage, the adaptive learning rate has an undesirably large variance, which leads to suspicious/bad local optima.*

To begin with, we first analyze a special case. When $t = 1$, we have $\psi(g_1) = \sqrt{\frac{1}{g_1^2}}$. We view $\{g_1, \cdots, g_t\}$ as i.i.d. random variables drawn from a Normal distribution $\mathcal{N}(0, \sigma^2)$[2]. Therefore, $\frac{1}{g_1^2}$ is subject to the scaled inverse chi-squared distribution, Scale-inv-$\mathcal{X}^2(1, \frac{1}{\sigma^2})$. Noted $\mathrm{Var}[\sqrt{\frac{1}{g_1^2}}] \propto \int_0^\infty x^{-1} e^{-x} dx$ and it is divergent. It means that the adaptive ratio can be undesirably large in the first stage of learning. Meanwhile, setting a small learning rate at the early stage can reduce the variance ($\mathrm{Var}[\alpha x] = \alpha^2 \mathrm{Var}[x]$), thus alleviate this problem. Therefore, we suggest it is the unbounded variance of the adaptive learning rate in the early stage that causes the problematic updates.

3.1 Warmup as Variance Reduction

In this section, we design a set of controlled experiments to verify our hypothesis. Particularly, we design two variants of Adam: *Adam-2k* and *Adam-eps*, and compare them to Adam with warmup and the vanilla Adam (without warmup) on the IWSLT'14 German to English dataset (Cettolo et al.). In the first two thousand iterations of Adam-2k, only the adaptive learning rate ($\psi(.)$) is updated, while the momentum ($\phi(.)$) and parameters (θ) are fixed[3]; other than this, it follows the original Adam algorithm. To comparison with other methods, its iterations are indexed from -1999 instead of 1. As in Figure 1, we observe that, after getting these additional two thousand samples for estimating the adaptive learning rate, Adam-2k avoids the convergence problem of the vanilla-Adam. Also, comparing Figure 2 and Figure 3, getting large enough samples prevents the gradient distribution from being distorted. These observations verify our hypothesis that the lack of sufficient data samples in the early stage is the root cause of the convergence issue.

[2] The mean zero normal assumption is valid at the beginning of the training, since weights are sampled from normal distributions with mean zero (Balduzzi et al., 2017), further analysis is conducted in Section 5.3.

[3] Different from Gotmare et al. (2019), all parameters and first moments are frozen in the first 2000 iterations.

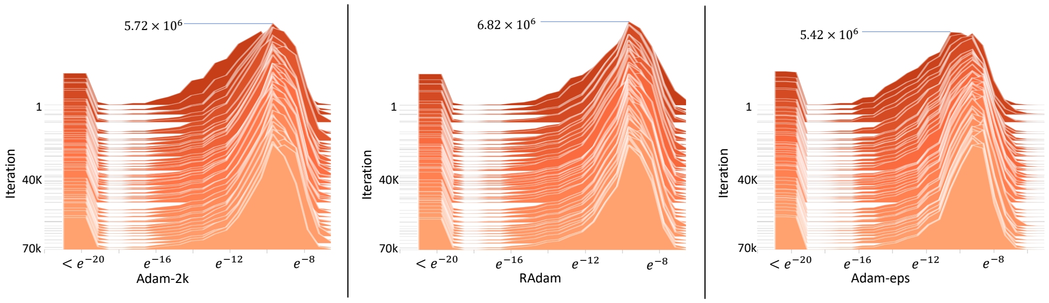


Figure 3: The histogram of the absolute value of gradients (on a log scale) during the training of Transformers on the De-En IWSLT' 14 dataset.

We further demonstrate that the convergence problem can be avoided by reducing the variance of the adaptive learning rate. A straightforward way to reduce the variance is to increase the value of ϵ in $\widehat{\psi}(g_1, \cdots, g_t) = \frac{\sqrt{1-\beta_2^t}}{\epsilon+\sqrt{(1-\beta_2)\sum_{i=1}^t \beta_2^{t-i} g_i^2}}$. Actually, if we assume $\widehat{\psi}(.)$ is subject to the uniform distribution, its variance equals to $\frac{1}{12\epsilon^2}$. Therefore, we design Adam-eps, which sets ϵ from a negligible value (1×10^{-8}) to a non-negligible value (1×10^{-4}). Its performance is summarized in Figure 1. We observe that it does not suffer from the serious convergence problem of vanilla-Adam. This demonstrates that the convergence problem can be alleviated by reducing the variance of the adaptive learning rate, and also explains why tuning ϵ is important in practice (Liu et al., 2019). Besides, similar to Adam-2k, it prevents the gradient distribution from being distorted (as shown in Figure 3). However, as in Figure 1, it produces a much worse performance comparing to Adam-2k and Adam-warmup. We conjecture that this is because large ϵ induces a large bias into the adaptive learning rate and slows down the optimization process. Thus, we need a more principled and rigorous way to control the variance of the adaptive learning rate. In the next subsection, we will present a theoretical analysis of the variance of the adaptive learning rate.

3.2 Analysis of Adaptive Learning Rate Variance

As mentioned before, Adam uses the exponential moving average to calculate the adaptive learning rate. For gradients $\{g_1, \cdots, g_t\}$, their exponential moving average has a larger variance than their simple average. Also, in the early stage (t is small), the difference of the exponential weights of $\{g_1, \cdots, g_t\}$ is relatively small (up to $1 - \beta_2^{t-1}$). Therefore, for ease of analysis, we approximate the distribution of the exponential moving average as the distribution of the simple average (Nau, 2014), *i.e.*, $p(\psi(.)) = p(\sqrt{\frac{1-\beta_2^t}{(1-\beta_2)\sum_{i=1}^t \beta_2^{t-i} g_i^2}}) \approx p(\sqrt{\frac{t}{\sum_{i=1}^t g_i^2}})$. Since $g_i \sim \mathcal{N}(0, \sigma^2)$, we have $\frac{t}{\sum_{i=1}^t g_i^2} \sim$ Scale-inv-$\mathcal{X}^2(t, \frac{1}{\sigma^2})$. Therefore, we assume $\frac{1-\beta_2^t}{(1-\beta_2)\sum_{i=1}^t \beta_2^{t-i} g_i^2}$ also subjects to a scaled inverse chi-square distribution with ρ degrees of freedom (further analysis on this approximation is conducted in Section 5.3). Based on this assumption, we have $\mathrm{Var}[\psi^2(.)]$ and the PDF of $\psi^2(.)$. Now, we proceed to the analysis of its square root variance, *i.e.*, $\mathrm{Var}[\psi(.)]$.

Theorem 1. If $\psi^2(.) \sim$ Scale-inv-$\mathcal{X}^2(\rho, \frac{1}{\sigma^2})$, $\mathrm{Var}[\psi(.)]$ monotonically decreases as ρ increases.

Proof. For ease of notation, we refer $\psi^2(.)$ as x and $\frac{1}{\sigma^2}$ as τ^2. Thus, $x \sim$ Scale-inv-$\mathcal{X}^2(\rho, \tau^2)$ and:

$$p(x) = \frac{(\tau^2\rho/2)^{\rho/2}}{\Gamma(\rho/2)} \frac{\exp[\frac{-\rho\tau^2}{2x}]}{x^{1+\rho/2}} \quad \text{and} \quad \mathbb{E}[x] = \frac{\rho}{(\rho-2)\sigma^2} \; (\forall\, \rho > 2) \tag{2}$$

where $\Gamma(.)$ is the gamma function. Therefore, we have:

$$\mathbb{E}[\sqrt{x}] = \int_0^\infty \sqrt{x}\, p(x)\, dx = \frac{\tau\sqrt{\rho}\,\Gamma(\rho/2-1)}{\sqrt{2}\,\Gamma(\rho/2)} \; (\forall\, \rho > 4). \tag{3}$$

Based on Equation 2 and 3, for $\forall\, \rho > 4$, we have:

$$\mathrm{Var}[\psi(.)] = \mathrm{Var}[\sqrt{x}] = \mathbb{E}[x] - \mathbb{E}[\sqrt{x}]^2 = \tau^2(\frac{\rho}{\rho-2} - \frac{\rho\, 2^{2\rho-5}}{\pi}\mathcal{B}(\frac{\rho-1}{2}, \frac{\rho-1}{2})^2) \tag{4}$$

where $\mathcal{B}(.)$ is the Beta function. By analyzing the derivative of $\mathrm{Var}[\psi(.)]$, we know it monotonically decreases as ρ increases. The detailed derivation is elaborated in the Appendix A. □

Theorem 1 gives a qualitative analysis of the variance of the adaptive learning rate. It shows that, due to the lack of training samples in the early stage, $\mathrm{Var}[\psi(.)]$ is larger than the late stage (Figure 8). To rigorously constraint the variance, we perform a quantified analysis on $\mathrm{Var}[\psi(.)]$ by estimating the degree of freedoms ρ.

4 Rectified Adaptive Learning Rate

In the previous section, Equation 4 gives the analytic form of $\mathrm{Var}[\psi(.)]$, where ρ is the degree of freedoms. Here, we first give an estimation of ρ based on t to conduct a quantified analysis for $\mathrm{Var}[\psi(g_1, \cdots, g_t)]$, then we describe the design of the learning rate rectification, and compare it to the heuristic warmup strategies.

4.1 Estimation of ρ

As the exponential moving average (EMA) is widely used in economics, it is usually interpreted as an approximation to the simple moving average (SMA) (Nau, 2014), *i.e.*,

$$p\left(\frac{(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}g_i^2}{1-\beta_2^t}\right) \approx p\left(\frac{\sum_{i=1}^{f(t,\beta_2)} g_{t+1-i}^2}{f(t,\beta_2)}\right). \tag{5}$$

where $f(t, \beta_2)$ is the length of the SMA which allows the SMA has the same "center of mass" with the EMA. In other words, $f(t, \beta_2)$ satisfies:

$$\frac{(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}(t+1-i)}{1-\beta_2^t} = \frac{\sum_{i=1}^{f(t,\beta_2)}(t+1-i)}{f(t,\beta_2)}.$$

By solving this equation, we have: $f(t, \beta_2) = \frac{2}{1-\beta_2} - 1 - \frac{2t\beta_2^t}{1-\beta_2^t}$. In the previous section, we assume: $\frac{1-\beta_2^t}{(1-\beta_2)\sum_{i=1}^{t}\beta_2^{t-i}g_i^2} \sim \text{Scale-inv-}\mathcal{X}^2(\rho, \frac{1}{\sigma^2})$. Here, since $g_i \sim \mathcal{N}(0, \sigma^2)$, we have $\frac{\sum_{i=1}^{f(t,\beta_2)} g_{t+1-i}^2}{f(t,\beta_2)} \sim \text{Scale-inv-}\mathcal{X}^2(f(t,\beta_2), \frac{1}{\sigma^2})$. Thus, Equation 5 views $\text{Scale-inv-}\mathcal{X}^2(f(t,\beta_2), \frac{1}{\sigma^2})$ as an approximation to $\text{Scale-inv-}\mathcal{X}^2(\rho, \frac{1}{\sigma^2})$. Therefore, we treat $f(t, \beta_2)$ as an estimation of ρ. For ease of notation, we mark $f(t, \beta_2)$ as ρ_t. Also, we record $\frac{2}{1-\beta_2} - 1$ as ρ_∞ (maximum length of the approximated SMA), due to the inequality $f(t, \beta_2) \leq \lim_{t\to\infty} f(t, \beta_2) = \frac{2}{1-\beta_2} - 1$.

4.2 Variance Estimation and Rectification

Based on the previous estimation, we have $\mathrm{Var}[\psi(.)] = \tau^2(\frac{\rho_t}{\rho_t - 2} - \frac{\rho_t\, 2^{2\rho_t-5}}{\pi}\mathcal{B}(\frac{\rho_t-1}{2}, \frac{\rho_t-1}{2})^2)$. The value of this function in the early stage is significantly larger than the late stage (as analyzed later, it decays roughly at the speed of $O(\frac{1}{\rho_t})$). For example, the variance at $\rho_t = 5$ is over 100 times larger than the variance at $\rho_t = 500$. Additionally, based on Theorem 1, we know $\min_{\rho_t} \mathrm{Var}[\psi(.)] = \mathrm{Var}[\psi(.)]|_{\rho_t=\rho_\infty}$ and mark this minimal value as C_{var}. In order to ensure that the adaptive learning rate ($\psi(.)$) has consistent variance, we rectify the variance at the t-th timestamp as below,

$$\mathrm{Var}[r_t\,\psi(g_1,\cdots,g_t)] = C_{\text{var}} \quad \text{where} \quad r_t = \sqrt{\frac{C_{\text{var}}}{\mathrm{Var}[\psi(g_1,\cdots,g_t)]}}.$$

Although we have the analytic form of $\mathrm{Var}[\psi(.)]$ (*i.e.*, Equation 4), it is not numerically stable. Therefore, we use the first-order approximation to calculate the rectification term. Specifically, by approximating $\sqrt{\psi^2(.)}$ to the first order (Wolter, 2007),

$$\sqrt{\psi^2(.)} \approx \sqrt{\mathbb{E}[\psi^2(.)]} + \frac{1}{2\sqrt{\mathbb{E}[\psi^2(.)]}}(\psi^2(.) - \mathbb{E}[\psi^2(.)]) \quad \text{and} \quad \mathrm{Var}[\psi(.)] \approx \frac{\mathrm{Var}[\psi^2(.)]}{4\,\mathbb{E}[\psi^2(.)]}.$$

Algorithm 2: Rectified Adam. All operations are element-wise.

```
Input: $\{\alpha_t\}_{t=1}^T$: step size, $\{\beta_1, \beta_2\}$: decay rate to calculate moving average and moving 2nd
        moment, $\theta_0$: initial parameter, $f_t(\theta)$: stochastic objective function.
Output: $\theta_t$: resulting parameters
$m_0, v_0 \leftarrow 0, 0$ (Initialize moving 1st and 2nd moment)
$\rho_\infty \leftarrow 2/(1-\beta_2) - 1$ (Compute the maximum length of the approximated SMA)
while $t = \{1, \cdots, T\}$ do
    $g_t \leftarrow \Delta_\theta f_t(\theta_{t-1})$ (Calculate gradients w.r.t. stochastic objective at timestep t)
    $v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$ (Update exponential moving 2nd moment)
    $m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$ (Update exponential moving 1st moment)
    $\widehat{m_t} \leftarrow m_t/(1-\beta_1^t)$ (Compute bias-corrected moving average)
    $\rho_t \leftarrow \rho_\infty - 2t\beta_2^t/(1-\beta_2^t)$(Compute the length of the approximated SMA)
    if the variance is tractable, i.e., $\rho_t > 4$ then
        $\widehat{v_t} \leftarrow \sqrt{v_t/(1-\beta_2^t)}$ (Compute bias-corrected moving 2nd moment)
        $r_t \leftarrow \sqrt{\frac{(\rho_t-4)(\rho_t-2)\rho_\infty}{(\rho_\infty-4)(\rho_\infty-2)\rho_t}}$ (Compute the variance rectification term)
        $\theta_t \leftarrow \theta_{t-1} - \alpha_t r_t \widehat{m_t}/\widehat{v_t}$ (Update parameters with adaptive momentum)
    else
        $\theta_t \leftarrow \theta_{t-1} - \alpha_t \widehat{m_t}$ (Update parameters with un-adapted momentum)
return $\theta_T$
```

Since $\psi^2(.) \sim$ Scale-inv-$\mathcal{X}^2(\rho_t, \frac{1}{\sigma^2})$, we have:

$$\mathrm{Var}[\psi(.)] \approx \frac{\rho_t}{2(\rho_t-2)(\rho_t-4)\sigma^2}. \tag{6}$$

In Section 5.3, we conduct simulation experiments to examine Equation 6 and find that it is a reliable approximation. Also, we know that $\mathrm{Var}[\sqrt{\psi(.)}]$ decreases approximately at the speed of $O(\frac{1}{\rho_t})$. With this approximation, we can calculate the rectification term as:

$$r_t = \sqrt{\frac{(\rho_t-4)(\rho_t-2)\rho_\infty}{(\rho_\infty-4)(\rho_\infty-2)\rho_t}}.$$

Applying our rectification term to Adam, we come up with a new variant of Adam, RAdam, as summarized in Algorithm 2. Specifically, when the length of the approximated SMA is less or equal than 4, the variance of the adaptive learning rate is intractable and the adaptive learning rate is inactivated. Otherwise, we calculate the variance rectification term and update parameters with the adaptive learning rate. It is worth mentioning that, if $\beta_2 \leq 0.6$, we have $\rho_\infty \leq 4$ and RAdam is degenerated to SGD with momentum.

4.3 In Comparison with Warmup

We notice that r_t has a similar form to the heuristic linear warmup, which can be viewed as setting the rectification term as $\frac{min(t, T_w)}{T_w}$. It verifies our intuition that warmup works as a variance reduction technique. Comparing these two strategies, RAdam deactivates the adaptive learning rate when its variance is divergent, thus avoiding undesired instability in the first few updates. Besides, our method does not require an additional hyperparameter (*i.e.*, T_w) to control the variance reduction and can automatically adapt to different moving average rules.

In this paper, we identify and fix an underlying issue of adaptive optimization methods instead of neural architectures. Thus, the proposed rectification term is orthogonal to other training stabilization techniques such as gradient clipping (Bengio et al., 2013), initialization (Balduzzi et al., 2017; Zhang et al., 2019) and normalization (Ba et al., 2016; Ioffe & Szegedy, 2015). Indeed, these techniques can be integrated with our proposed variance rectification. Specifically, since warmup is originally proposed to handle *gradient variance* for SGD (Goyal et al., 2017; Gotmare et al., 2019; Xiao et al., 2019), RAdam can also be integrated with the warmup heuristic to handle some extreme cases such as training with very large batches.

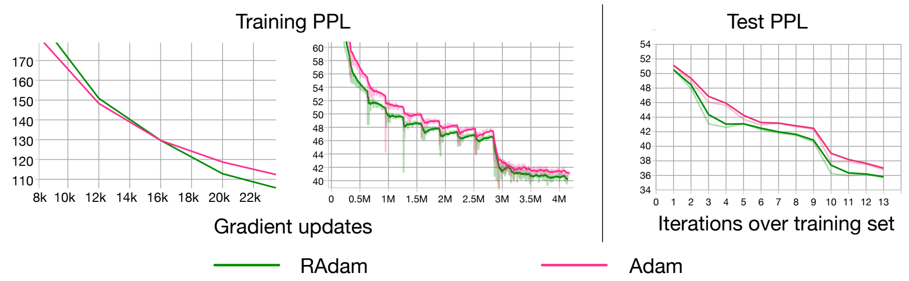


Figure 4: Language modeling (LSTMs) training on the One Billion Word dataset.

5 Experiments

We evaluate RAdam on several benchmarks[4]: One Billion Word for Language Modeling; Cifar10 and ImageNet for Image Classification. Following Loshchilov & Hutter (2017), we decouple weight decays in the vanilla Adam, Adam with warmup and RAdam in our experiments.

5.1 Comparing to Vanilla Adam

As analyzed before, the adaptive learning rate has undesirably large variance in the early stage of training and leads to suspicious/bad local optima on NMT. One question we are interested in answering is: whether such an issue widely exits in other similar tasks and applications. Thus, we conduct a set of experiments with two classical tasks of NLP and CV, *i.e.*, language modeling and image classification. RAdam not only results in consistent improvements over the vanilla Adam, but also demonstrates its robustness to the change of learning rates. It verifies that the variance issue exists in various machine learning applications, and has a big impact on the model behavior. Detailed comparison and analysis are described as follows.

Table 1: Perplexity on Language Modeling

Method	One Billion Word
Adam	36.92
RAdam	35.70

Table 2: Accuracy on Image Classification

Method	CIFAR10	ImageNet
SGD	91.51	69.86
Adam	90.54	66.54
RAdam	91.38	67.62

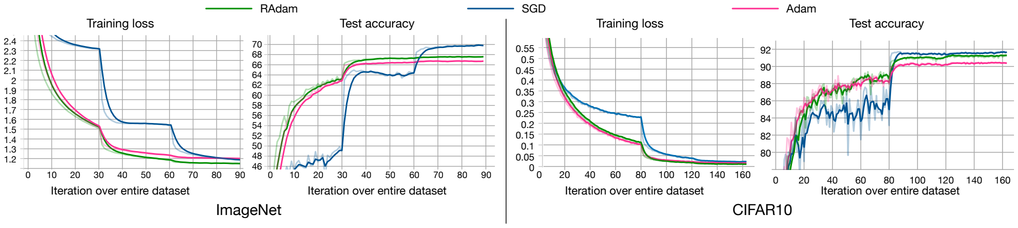


Figure 5: Training of ResNet-18 on the ImageNet and ResNet-20 on the CIFAR10 dataset.

Performance Comparison. The performances on language modeling (*i.e.*, One Billion Word [5] (Chelba et al., 2013)) and image classification (*i.e.*, CIFAR10 (Krizhevsky et al., 2009) and ImageNet (Deng et al., 2009)) are summarized in Table 1 and Table 2, and their learning curves are presented in Figure 4 and Figure 5, respectively. The results show that RAdam outperforms Adam in all three datasets. As shown in Figure 4, although the rectification term makes RAdam slower than the vanilla Adam in the first few epochs, it allows RAdam to converge faster after that. In other words, by reducing the variance of the adaptive learning rate in the early stage, it gets both faster convergence and better performance, which verifies the impact of the variance issue. We also observe that RAdam obtains consistent improvements over Adam on image classification. It is worth

[4] The detailed hyperparameter settings are elaborated in the Appendix B

[5] Rare words that occur less than 3 times are replaced with a special token, the resulting dictionary is shrank from 7.9M to 6.4M.

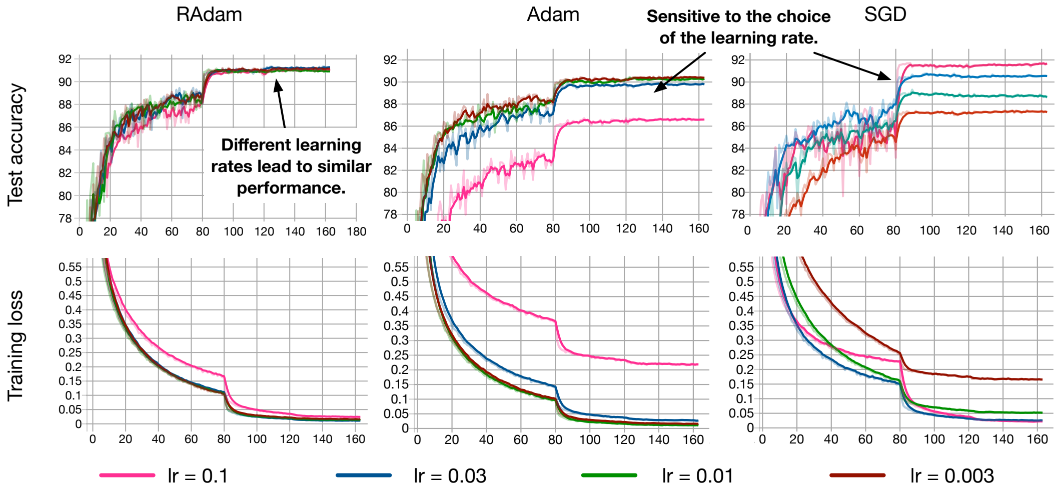


Figure 6: Performance of RAdam, Adam and SGD with different learning rates on CIFAR10. X-axis is the number of epochs.

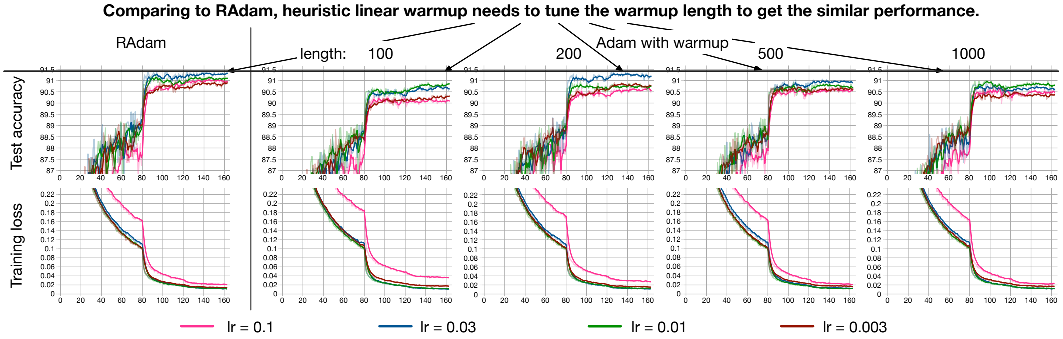


Figure 7: Performance of RAdam, Adam with warmup on CIFAR10 with different learning rates. X-axis is the number of epochs.

noting that, on both ImageNet and CIFAR10, although RAdam fails to outperform SGD in terms of test accuracy, it results in a better training performance (*e.g.*, the training accuracy of SGD, Adam, and RAdam on ImageNet are 69.57, 69.12 and 70.30 respectively).

Robustness to Learning Rate Change. Besides performance improvements, RAdam also improves the robustness of model training. We use different initial learning rates, conduct experiments with ResNet-20 on the CIFAR10 datasets, and summarize their performance in Figure 6. For learning rates within a broad range (*i.e.*, $\{0.1, 0.03, 0.01, 0.003\}$), RAdam achieves consistent model performances (their test accuracy curves highly overlap with each other), while Adam and SGD are shown to be sensitive to the learning rate. The observation can be interpreted that by rectifying the variance of the adaptive learning rate, RAdam improves the robustness of model training and can adapt to different learning rates of a broader range.

5.2 Comparing to Heuristic Warmup

To examine the effectiveness of RAdam, we first conduct comparisons on neural machine translation, on which the state-of-the-art employs Adam with the linear warmup. Specifically, we conduct experiments on three datasets, i.e., IWSLT'14 De-En, IWSLT'14 En-De, and WMT'16 En-De. Due to the limited size of the IWSLT'14 dataset, we conduct experiments using 5 different random seeds and report their mean and standard derivation. As discussed before, the vanilla Adam algorithm leads to suspicious/bad local optima (i.e., converges to a training perplexity around 500), and needs a learning rate warmup stage to stabilize the training.

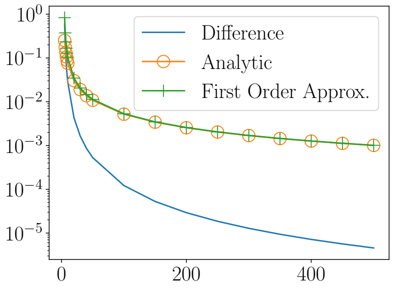
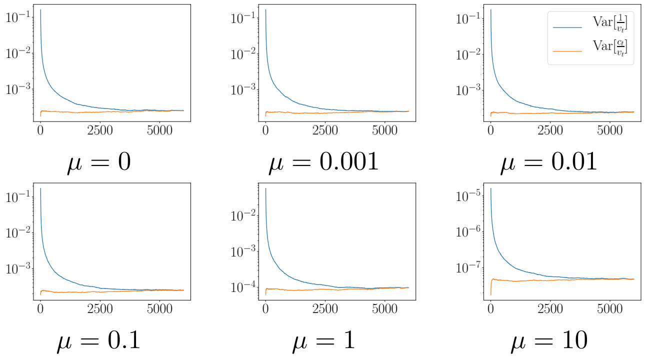


Figure 8: The value of Equation 4, Equation 6 and their difference (calculated as the absolute difference value). The x-axis is ρ and the y-axis is the variance in the log scale.



$\mu = 0$ $\mu = 0.001$ $\mu = 0.01$

$\mu = 0.1$ $\mu = 1$ $\mu = 10$

Figure 9: The simulation of $\text{Var}[\frac{1}{v_t}]$ and $\text{Var}[\frac{c_t}{v_t}]$. The x-axis is iteration number (the simulation starts from 5) and the y-axis is the variance in the log scale.

We summarize the performance obtained with the heuristic warmup and our proposed rectification term in Table 3 and visualize the training curve of IWSLT De-En in Figure 1. With a consistent adaptive learning rate variance, our proposed method achieves similar performance to that of previous state-of-the-art warmup heuristics. It verifies our intuition that the problematic updates of Adam are indeed caused by the undesirably large variance in the early stage.

Table 3: BLEU score on Neural Machine Translation.

Method	IWSLT'14 DE-EN	IWSLT'14 EN-DE	WMT'16 EN-DE
Adam with warmup	34.66 ± 0.014	28.56 ± 0.067	27.03
RAdam	34.76 ± 0.003	28.48 ± 0.054	27.27

Moreover, we applied Adam with warmup on the CIFAR10 dataset. Its best accuracy on the test set is 91.29, which is similar to RAdam (91.38). However, we found that RAdam requires less hyperparameter tuning. Specifically, we visualize their learning curves in Figure 7. For some warmup steps, Adam with warmup is relatively more sensitive to the choice of the learning rate. RAdam, at the same time, is not only more robust, but also can automatically control the warmup behavior (*i.e.*, without requiring the length of warmup). For example, when setting the learning rate as 0.1, Adam with 100 steps of warmup fails to get satisfying performance and only results in an accuracy of 90.13; RAdam successfully gets a accuracy of 91.06, with the original setting of the moving average calculation (*i.e.*, $\beta_1 = 0.9, \beta_2 = 0.999$). We conjecture the reason is due to the fact that RAdam, which is based on a rigorous variance analysis, explicitly avoids the extreme situation where the variance is divergent, and rectifies the variance to be consistent in other situations.

5.3 Simulated Verification

In Sections 3 and 4, we approximate $\text{Var}[\sqrt{\frac{t}{\sum_{i=1}^t g_i^2}}]$ to the first order, and assume $\psi^2(.) = \frac{1-\beta_2^t}{(1-\beta_2)\sum_{i=1}^t \beta_2^{t-i} g_i^2}$ subjects to a scaled inverse chi-square distribution (covers the approximation from EMA to SMA). In this section, we examine these two approximations using simulations.

First Order Approximation of $\text{Var}[\sqrt{\frac{t}{\sum_{i=1}^t g_i^2}}]$. To compare Equations 6 and 4, we assume $\tau = 1$ and plot their values and their difference for $\nu = \{5, \cdots, 500\}$ in Figure 8. The curve of the analytic form and the first-order approximation highly overlap, and their difference is much smaller than their value by more than an order of magnitude. This result verifies the reliability of our first-order approximation.

Scaled Inverse Chi-Square Distribution Assumption. In this paper, we assume g_i accords to a Normal distribution with a zero mean. We also assume $\psi^2(.)$ accords to the scaled inverse chi-square distribution to derive the variance of $\mathrm{Var}[\psi(.)]$, based on the similarity between the exponential moving average and simple moving average. Here, we empirically verify this assumption.

Specifically, since g_i in the optimization problem may not be zero-mean, we assume its expectation is μ and sample g_i from $\mathcal{N}(\mu, 1)$. Then, based on these samples, we calculate the variance of the original adaptive learning rate and the proposed rectified adaptive learning rate, *i.e.*, $\mathrm{Var}[\frac{1}{\hat{v}_t}]$ and $\mathrm{Var}[\frac{r_t}{\hat{v}_t}]$ respectively. We set β_2 to 0.999, the number of sampled trajectories to 5000, the number of iterations to 6000, and summarize the simulation results in Figure 9. Across all six settings with different μ, the adaptive learning rate has a larger variance in the first stage and the rectified adaptive learning rate has relative consistent variance. This verifies the reliability of our assumption.

6 Conclusion

In this paper, we explore the underlying principle of the effectiveness of the warmup heuristic used for adaptive optimization algorithms. Specifically, we identify that, due to the limited amount of samples in the early stage of model training, the adaptive learning rate has an undesirably large variance and can cause the model to converge to suspicious/bad local optima. We provide both empirical and theoretical evidence to support our hypothesis, and further propose a new variant of Adam, whose adaptive learning rate is rectified so as to have a consistent variance. Empirical results demonstrate the effectiveness of our proposed method. In future work, we plan to apply the proposed method to other applications such as Named Entity Recognition (Reimers & Gurevych, 2017; Lin et al., 2019). Another interesting direction to pursue is to adapt the choice of β based on the variance estimation of different parameters, *i.e.*, use a larger β for parameters with a larger variance.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 342–350. JMLR. org, 2017.

Yoshua Bengio, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. Advances in optimizing recurrent networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8624–8628. IEEE, 2013.

Augustin Cauchy. Méthode générale pour la résolution des systemes déquations simultanées. *Comp. Rend. Sci. Paris*, 25(1847):536–538, 1847.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th iwslt evaluation campaign, iwslt 2014.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Timothy Dozat. Incorporating nesterov momentum into adam. 2016.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Akhilesh Gotmare, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. A closer look at deep learning heuristics: Learning rate restarts, warmup and distillation. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=r14EOsCqKX`.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Ying Lin, Liyuan Liu, Heng Ji, Dong Yu, and Jiawei Han. Reliability-aware dynamic feature composition for name tagging. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pp. 165–174, 2019.

Liyuan Liu, Xiang Ren, Jingbo Shang, Jian Peng, and Jiawei Han. Efficient contextualized representation: Language model pruning for sequence labeling. *arXiv preprint arXiv:1804.07827*, 2018.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 2017.

Robert Nau. Forecasting with moving averages. 2014.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*, 2019.

Martin Popel and Ondřej Bojar. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70, 2018.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

Nils Reimers and Iryna Gurevych. Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. *arXiv preprint arXiv:1707.06799*, 2017.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop, coursera: Neural networks for machine learning. *University of Toronto, Technical Report*, 2012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Kirk M Wolter. Taylor series methods. In *Introduction to variance estimation*, pp. 226–271. Springer, 2007.

Lin Xiao, Adams Wei Yu, Qihang Lin, and Weizhu Chen. Dscovr: Randomized primal-dual block coordinate algorithms for asynchronous distributed optimization. *Journal of Machine Learning Research*, 20(43):1–58, 2019.

Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Hongyi Zhang, Yann N Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. *arXiv preprint arXiv:1901.09321*, 2019.

A PROOF OF THEOREM 1

To prove the monotonic, we need to show

Lemma 1. for $t \geq 4$, $\frac{\partial}{\partial t}(\frac{t}{t-2} - \frac{t\,2^{2t-5}}{\pi}\mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2) < 0$

Proof. The target inequality can be re-wrote as

$$\begin{aligned}
&\frac{\partial}{\partial t}(\frac{t}{t-2} - \frac{t\,2^{2t-5}}{\pi}\mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2) \\
&= \frac{-2}{(t-2)^2} - \frac{2^{2t-5}}{\pi}\mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2 - \frac{t\,2^{2t-5}\ln 4}{\pi}\mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2 \\
&- \frac{2t\,2^{2t-5}}{\pi}\mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2(\Psi(\frac{t-1}{2}) - \Psi(t-1)), \quad \left(\Psi(x) = \frac{\Gamma'(x)}{\Gamma(x)}\right) \\
&< 0
\end{aligned}$$

This inequality is equivalent to:

$$\begin{aligned}
\frac{64\pi}{(t-2)^2 4^t \mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2} + 1 + t\ln 4 + 2t\Psi(\frac{t-1}{2}) \\
\geq 2t\Psi(t-1) \overset{(i)}{=} t[\Psi(\frac{t-1}{2}) + \Psi(\frac{t}{2}) + \ln 4],
\end{aligned}$$

where (i) is derived from Legendre duplication formula. Simplify the above inequality, we get:

$$\frac{64\pi}{(t-2)^2 4^t \mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2} + 1 + t\Psi(\frac{t-1}{2}) - t\Psi(\frac{t}{2}) \geq 0,$$

We only need to show

$$\begin{aligned}
&\frac{64\pi}{(t-2)^2 4^t \mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2} + 1 + t\Psi(\frac{t-1}{2}) - t\Psi(\frac{t}{2}) \\
&\geq \frac{64\pi}{(t-2)^2 4^t \mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2} + 2 + t(\ln(t/2) - 1/(t/2 - 0.5)) - t\ln(t/2) \\
&= \frac{64\pi}{(t-2)^2 4^t \mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2} - \frac{2}{t-1} \\
&> \frac{64\pi}{(t-2)^2 4^t \mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2} - \frac{2}{t-2} \geq 0,
\end{aligned}$$

where the first inequality is from $\ln(x) - 1/(2x) > \Psi(x) > \ln(x + 0.5) - 1/x$.

Therefore, we only need to show

$$32\pi \geq (t-2)4^t\mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2,$$

which is equivalent to

$$\begin{aligned}
(t-2)4^t\mathcal{B}(\frac{t-1}{2}, \frac{t-1}{2})^2 &= (t-2)4^t\frac{\Gamma(\frac{t-1}{2})^4}{\Gamma(t-1)^2} \\
&\overset{(i)}{=} (t-2)4^t\frac{\Gamma(\frac{t-1}{2})^2}{\Gamma(t/2)^2}4^{2-t}\pi = 16\pi(t-2)\frac{\Gamma(\frac{t-1}{2})^2}{\Gamma(t/2)^2} \leq 32\pi,
\end{aligned}$$

where (i) is from Legendre duplication formula.

So we only need to show

$$(t-2)\frac{\Gamma(\frac{t-1}{2})^2}{\Gamma(t/2)^2} \leq 2 \tag{7}$$

Using Gautschi's inequality ($\frac{\Gamma(x+1)}{\Gamma(x+s)} < (x+1)^{1-s}$), we have

$$(t-2)\frac{\Gamma(\frac{t-1}{2})^2}{\Gamma(t/2)^2} \leq (t-2)(\frac{t-1}{2})^{-1} = \frac{2(t-2)}{t-1} < 2 \tag{8}$$

□

B Implementation Details

B.1 Language Modeling

Our implementation is based on the previous work (Liu et al., 2018). Specifically, we use two-layer LSTMs with 2048 hidden states with adaptive softmax to conduct experiments on the one billion words dataset. Word embedding (random initialized) of 300 dimensions is used as the input and the adaptive softmax is incorporated with a default setting (cut-offs are set to $[4000, 40000, 200000]$). Additionally, as pre-processing, we replace all tokens occurring equal or less than 3 times with as UNK, which shrinks the dictionary from 7.9M to 6.4M. Dropout is applied to each layer with a ratio of 0.1, gradients are clipped at 5.0. We use the default hyper-parameters to update moving averages, *i.e.*$\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate is set to start from 0.001, and decayed at the start of 10th epochs. LSTMs are unrolled for 20 steps without resetting the LSTM states and the batch size is set to 128. All models are trained on one NVIDIA Tesla V100 GPU.

B.2 Imageine Classification

We use the default ResNet architectures (He et al., 2016) in a public pytorch re-implementation[6]. Specifically, we use 20-layer ResNet (9 Basic Blocks) for CIFAR-10 and 18-layer ResNet (8 Basic Blocks) for ImageNet. Batch size is 128 for CIFAR-10 and 256 for ImageNet. The model is trained for 186 epoches and the learning rate decays at the 81-th and the 122-th epoches by 0.1 on CIFAR-10, while the model is trained for 90 epoches and the learning rate decays at the 31-th and the 61-th epoch by 0.1 on ImageNet. For Adam and RAdam, we set $\beta_1 = 0.9, \beta_2 = 0.999$. For SGD, we set the momentum factor as 0.9. The weight decay rate is 10^{-4}. Random cropping and random horizontal flipping are applied to training data.

B.3 Neural Machine Translation

Our experiments are based on the default Transformers (Vaswani et al., 2017) implementation from the fairseq package (Ott et al., 2019). Specifically, we use word embedding with 512 dimensions and 6-layer encoder / decoder with 4 head and 1024 hidden dimensions on the IWSLT14' dataset; use word embedding with 512 dimension and 6-layer encoder / decoder with 8 heads and 2048 hidden dimensions. Label smoothed cross entropy is used as the objective function with an uncertainty $=$ 0.1 (Szegedy et al., 2016). We use linear learning rate decay starting from $3e^{-4}$, and the checkpoints of the last 20 epoches are averaged before evaluation. As to the wamrup strategy, we use a linear warmup for Adam in the first 4000 updates, and set β_2 to satisfy $\nu = 4000$ ($\beta_2 = 0.9995$). In the IWSLT'14 dataset, we conduct training on one NVIDIA Tesla V100 GPU, set maximum batch size as 4000, apply dropout with a ratio 0.3, using weight decay of 0.0001 and clip the gradient norm at 25. In the WMT'16 dataset, we conduct training on four NVIDIA Quadro R8000 GPUs and set maximum batch size as 8196.

[6] `https://github.com/bearpaw/pytorch-classification`